BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 February 2002 (28.02.2002)

PCT

(10) International Publication Number
WO 02/17139 A1

(51) International Patent Classification[7]: G06F 17/30

(21) International Application Number: PCT/US01/25491

(22) International Filing Date: 15 August 2001 (15.08.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/226,405 18 August 2000 (18.08.2000) US

(71) Applicant *(for all designated States except US)*: **AKAMAI TECHNOLOGIES, INC.** [US/US]; 500 Technology Square, Cambridge, MA 02139 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **DHANIDINA, Rizwan, S.** [US/US]; 169 Msgr. O'Brien Highway #816, Cambridge, MA 02141 (US). **KLONINGER, John, Josef** [US/US]; 6 Magazine Court, Cambridge, MA 02139 (US). **ROSE, Kyle** [US/US]; 6 Russell Road, Somerville, MA 02144 (US). **SUNDARAM, Ravi** [IN/IN]; 76 Chilton Street, Cambridge, MA 02138 (US). **YERUSHALMI, Yoav** [IL/IL]; 50 Boston Street, Somerville, MA 02143 (US).

(74) Agent: **JUDSON, David, H.**; Akamai Technologies, Inc., 500 Technology Square, Cambridge, MA 02139 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

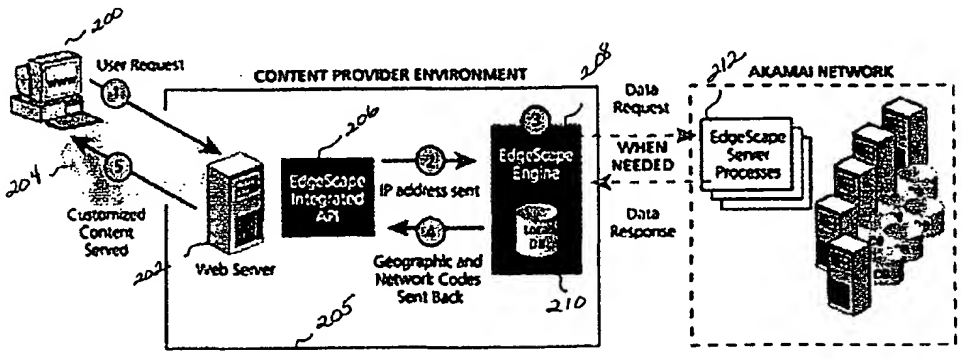
(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declaration under Rule 4.17:
— *of inventorship (Rule 4.17(iv)) for US only*

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

[Continued on next page]

(54) Title: METHOD AND SYSTEM FOR PROVIDING CONTENT PROVIDERS WITH INFORMATION ABOUT HOW THEIR USERS ACCESS THE INTERNET



(57) Abstract: A system and method for accurately identifying the geographic location from which users access a Web site and the network origin of a user's request. The end user (200) interacts with Web server (202) over network (204) which may be the Internet. Web server (202) is provisioned with Applicaion Program Interface API (206), engine (208) and a local instance of the database (210) within the content provider execution environment (205). The content delivery network CDN manages database update processes (212). In operation, the user request for the Web site reaches the Web sites reaches the Web server (202) which uses the API (206) to send the user's IP address to the engine (208). If the engine (208) does not have the user's geographic and network data stored locally in the database (210) or cannot obtain it, the engine (208) retrieves that data form the knowledge base sotred on the CDN.

WO 02/17139 A1

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD AND SYSTEM FOR PROVIDING CONTENT PROVIDERS WITH INFORMATION ABOUT HOW THEIR USERS ACCESS THE INTERNET

This application is based on and claims priority from Provisional Application Serial No. 60/226,405, filed August 18, 2000.

BACKGROUND OF THE INVENTION

Technical Field

The present invention relates generally to knowledge delivery systems that enable content providers to more intelligently serve content and control assets on their Web sites.

Description of the Related Art

As the Internet has matured and users' expectations have grown for efficient and direct access to the information they seek, content providers face the challenge of implementing functionality on their Web sites that selectively presents content tailored to each customer's unique needs. The days of endlessly clicking through layers of links on Web sites designed in hierarchical categories with a one-size-fits-all mentality are long gone. The trend today is toward serving dynamically-generated content based on predetermined knowledge of the user visiting a site. Personalization technology is intended to customize the delivery of content to users so their experience on a Web site has more relevance and efficiency each time they visit. This personalized treatment serves to increase the frequency of visits to a particular Web site and strengthen user retention rates.

Having information about users on the Web is not an exercise in guesswork, but rather a mixture of technologies and processes that glean information from network operations or that take advantage of user cooperation in supplying demographic information. Information gathering generally falls into two categories: user-declared information and anonymous information gathering. The former is the most effective way to learn about user preferences and demographics, but it is often burdened by signficant levels of inaccuracy due to false user information submission or an incomplete collection of profiles compared to the entire Internet population. Anonymous information gathering is more comprehensive in that it can encomass the analysis of a great amount, if not all, of the Internet without being dependent upon user input. While it cannot offer the same specificity of user information gained when users disclose information themselves, this technique can generate comprehensive information about a user's Internet access point, which can offer a great deal of insight toward delivering content specific to the user's environment.

Thus, it would be highly desirable to be able to provide Internet content providers with up-to-date information about how end users access their Web sites. The scope of the problem, however, is immense. It is estimated that the Internet itself presently has 4-5 billion Internet Protocol (IP) addresses, and those addresses are grouped into approximately 300,000 CIDR (Classless InterDomain Routing) blocks. As is well known, Internet access points undergo constant change due to the rapid growth, technological evolutions and individual network maintenance preferences by companies and network providers all over the world. ISPs and businesses are assigned IP blocks, however, these entities make their own decisions about how to assign the IP addresses associated with a given IP block. These IP assignments also are constantly changing.

Well-managed content delivery networks (CDNs) constantly monitor and map the Internet and the billions of IP addresses that connect all of the computers and users operating on the Web everyday. As a consequence, a CDN has the ability to gather, verify and enhance information about how end users access the Internet. There remains a need in the art, however, to provide an information delivery system that enables content providers to access that information in an highly available, scalable and efficient manner. The present invention addresses this problem.

BRIEF SUMMARY OF THE INVENTION

The present invention provides an information access and retrieval method, system and/or managed service for accurately identifying, among other access attributes, the geographic location from which users access a Web site and the network origin of a user's request. In an illustrative embodiment, the invention is implemented as an information retrieval system within or as an adjunct to a content provider application platform (e.g., a Web server, an application server, or the like) to provide geo-approximation and bandwidth characterization with respect to a given end user access request. Thus, for example, for each user visiting a content provider's Web site, the inventive functionality delivers given metrics, for example: an ISO standard two-letter country code of the request origin, a state or region code of countries, the network origin of a user's access, network type (e.g., dial-up, DSL, cable modem, etc.), and other metrics of interest including, without limitation, Designated Marketing Area (DMA), Metropolitan Statistical Area (MSA), Primary Metropolitan Statistical Area (PMSA), longitude, latitude, FIPS, time zone, and the like.

In one embodiment, the invention is a managed service that combines a software client with connectivity to a content delivery network (or other third party source) for receiving a database associating IP CIDR blocks with given access metrics. In this embodiment, content

providers implement the service by installing an application programming interface (API) and access "engine" directly onto a web or application server platform. An instance of the database preferably is stored locally at the application server platform. When a user visits the content provider's Web site, a query is made through the API to the engine. The query triggers a lookup in the database to associate the user's IP address to an appropriate metric, such as country code (and region code, etc.). The country code and other network information output can then be used for custom content functionality based on what applications the provider is using. The engine sets-up and maintains secure connectivity to one or more database update processes, which are preferably maintained and managed by a third party such as an Internet CDNSP. On a periodic basis, the engine downloads the latest mapping database from the CDN or other third party provider) to ensure that the data for the server is as up-to-date as possible.

The foregoing has outlined some of the pertinent features and advantages of the present invention. A more complete understanding of the invention is provided in the following Detailed Description of the Preferred Embodiment.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a simplified block diagram of the components of the information delivery system of the present invention;

Figure 2 is a simplified diagram illustrating how an end user request is processed at a Web server that has been provisioned with the information delivery system of the present invention; and

Figure 3 is a block diagram illustrating in more detail the components of the information delivery system of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

The information delivery system of the present invention preferably leverages off of end user access data compiled by an Internet content delivery network (ICDN). A conventional CDN is implemented as a combination of a content delivery infrastructure, a request-routing mechanism, and a distribution infrastructure. The content delivery infrastructure usually comprises a set of "surrogate" origin servers that are located at strategic locations (e.g., Internet network access points, and the like) for delivering copies of content to requesting end users. The request-routing mechanism allocates servers in the content delivery infrastructure to requesting clients in a way that, for web content delivery, minimizes a given client's response time and, for streaming media delivery, provides for the highest quality. The distribution infrastructure consists of on-demand or push-based mechanisms that move content from the origin server to the surrogates. A CDN service provider (CDNSP) may organize sets of surrogate origin servers as a

"region." In this type of arrangement, an ICDN region typically comprises a set of one or more content servers that share a common backend, e.g., a LAN, and that are located at or near an Internet access point. Thus, for example, a typical ICDN region may be colocated within an Internet Service Provider Point of Presence (PoP). A representative ICDN content server is a Pentium-based caching appliance running an operating system (e.g., Linux, Windows NT, Windows 2000) and having suitable RAM and disk storage for ICDN applications and content delivery network content (e.g., HTTP content, streaming media and applications). The ICDN typically also includes network agents that monitor the network as well as the server loads. These network agents are typically collocated at third party data centers. Map maker software receives data generated from the network agents and periodically creates maps that dynamically associate IP addresses (e.g., the IP addresses of client-side local name servers) with the ICDN regions. In one type of service offering, known as Akamai FreeFlow, from Akamai Technologies, Inc. of Cambridge, Massachusetts, requests for content that has been tagged for delivery from the ICDN are directed to the "best" region (using a map-driven DNS request routing mechanism) and to a content server within the region that is not overloaded and that is likely to host the requested content.

A well-managed ICDN constantly monitors and maps the Internet and the billions of IP addresses that connect all of the computers and users operating on the Web everyday. As a consequence, the CDNSP has the ability to gather, verify and enhance information about how end users access the Internet. Using a variety of techniques to monitor, map, quantify and verify network status and positioning on an ongoing and global basis, the CDNSP populates and constantly updates a database that contains geo-location and other end user access information. For purposes of illustration, the database associates IP-to-geo-location data for the entire public Internet, or for any given portion thereof. IP address data in the database may be collected from various sources, e.g., CDN name server lookup entries, network provider data, sample IP address to geography resolutions generated from testing, public registration data including domain registries (e.g., APNIC, ARIN, RIPE databases), commercial third party information sources, and the like. Preferably, the database stores data about groups of IP addresses, although this is not a limitation. Thus, e.g., the database groupings associate user IP addresses and network information with CIDR blocks. The techniques for data assembly, compilation, and updating are outside the scope of the present invention. It is assumed, however, that the database is available to the information delivery system and is continually updated on a periodic basis (e.g., every few hours, every few days, or the like). The invention provides an information delivery system for access and retrieval of such database information by a content provider. The delivery system of

the invention may be implemented in whole or in part by a third party service provider (e.g., by the CDNSP) as a "managed" service offering or it may be implemented as a standalone "product."

According to the present invention, the database stores information associating IP address blocks with given geo- and network data including, without limitation, one or more of the following attributes: country code, region (e.g., state (e.g., US/non-AOL only) or province (e.g., Canada), city (e.g., US/with a radius of 50 miles), Designated Marketing Area (DMA), Metropolitan Statistical Area (MSA), Primary Metropolitan Statistical Area (PMSA), longitude, latitude, county, FIPS, time zone, origin network, network type, throughput, and the like. Preferably, the entire IP address space is represented in the database, and at least country code information is always associated with a valid IP address. If an IP address is in reserved IP space, preferably every attribute has a return value of "reserved." Preferably, all other elements in the database, including region code, are optional and may or may not be associated with an IP address. A representative portion of the database is illustrated below:

```
1.0.13.0/24:
        country_code=AU:network_type=isdn:network=att
1.0.15.126/31:
        country_code=JP:network_type=dialup:network=ibm
1.0.17.38/32:
        country_code=US:region_code=PA:city=PITTSBURGH:dma=508:msa=6280:pmsa=45:areacode=412:cou
nty=ALLEGHENY:fips=42003:lat=40.4338:long=79.8642:timezone=EST:network=ATT:network_type=dsl:throug
hput=low

1.0.25.0/24:
        country_code=MK
```

With reference now to Figure 1, the information delivery system 100 of the present invention preferably includes three (3) primary components: an application programming interface (API), an engine (sometimes called a Facilitator), and a database update process or daemon. Generally, each content provider uses an API and an engine, and a given engine can establish and maintain connections to one or more database update processes. One or more update processes can maintain communications with multiple engines, including engines from different content providers. The database may be deemed to be part of the information delivery system or simply an adjunct thereto.

As illustrated in Figure 1, the API 102 and engine 104 components of the information delivery system 100 are implemented within the execution environment of a content provider platform 108, such as a server. Platform 108 is assumed to be a system, machine, application, program or execution thread of a given application that has the capability of using geo- and/or

network data obtained from an instance 106 of the database, which is conveniently hosted in a cache 110 or other suitable storage of the application platform 108. Representative platforms include Web servers, application servers, database servers, legacy systems, and the like. The given application can be any functionality that can leverage individual user access as described in more detail below.

Preferably, the API is code that is integrated into the application and is used to make queries to the engine. The engine is responsible for responding to requests from the API and, in response thereto, retrieving the requested geo-data from the database and returning it to the API for delivery back to and use by the requesting application. The engine also operates as a daemon or system agent (i.e., a background task) that provides management of a connection between the API and the database update process, which itself may be a daemon or system agent. As also seen in Figure 1, multiple instances of the database daemon preferably execute on servers throughout the CDN so that a given engine has the ability to obtain the update information even if a particular update process fails or the connection between the engine and an update process fails. The CDN periodically generates the IP-to-geo-location maps that are transferred (e.g., via ftp) to the database update daemon(s) and, thus, to the engine(s) operated by the content providers.

Generalizing, the information delivery system of the present invention is implemented as software in commodity hardware running an operating system. Although the processes are shown as separate and distinct for illustrative and discussion purposes, given functionalities may be integrated into one or more processes, modules, routines, execution threads, or other known programming constructs. Thus, for example, the API may be suitably combined with the engine. One or more of the processes typically include other sub-processes or functions.

Figure 2 illustrates the basic operation of the information delivery system. It is assumed that end user 200 interacts with Web server 202 over network 204, which may be the Internet. Web server 202 has been provisioned with the API 206, the engine 208, and a local instance of the database 210 within the content provider execution environment 205. As described above, the content delivery network (CDN) manages the database update processes 212. In operation, the user request for the Web site reaches the Web server. This is step (1). At step (2), the Web server uses the integrated API 206 to send the user's IP address to the engine 208. If the engine does not have the user's geographic and network data stored locally in the database 210 or cannot otherwise obtain it, the engine retrieves that data from the knowledge base stored on the CDN. This is step (3). There may be many reasons why the engine cannot obtain the data locally. The data may not be present. The database itself may have been corrupted. The

machine on which the database resides may not have sufficient memory. Returning back to the drawing, it is assumed that the engine has obtained the data from either the local database or from a remote instance. At step (4), the engine 208 sends the user data back to the integrated API 206. At step (5), the Web server selects customized content for delivery to the user based on the geo-location and/or network data provided. The local instance of the database 210 is updated periodically to ensure that the Web server application has the most up-to-date information.

As illustrated in Figure 3, the invention (whether a managed service or a standalone product) preferably is architected using three (3) subsystems or components that interact with a Web site's application environment: ClientAPI 300, Facilitator 302 (the "engine"), and a Datad (the "database update") process 304. As noted above, ClientAPI is a piece of code that is integrated into a customer application to make queries to the service, specifically the Facilitator. The Facilitator is a process that runs in the customer environment and is responsible for establishing and managing connections between the customer application and the CDN, specifically one or more Datad processes. The Facilitator is also responsible for listening and responding to requests from the ClientAPI. The Datad is the process that preferably runs in multiple locations on the CDNSP's network. The primary purpose of the Datad component is to respond to queries from Facilitator processes in an authenticated, secure, and highly available way, and to manage updates to the database that is stored, preferably on the same servers as Datad processes. Preferably, database queries are serviced by the Facilitator locally if at all possible to avoid network transport, but this is not always possible for the reasons discussed above.

In a representative embodiment, ClientAPI 300 is source code that is integrated into the customer Web and/or application servers to make queries to the Facilitator. The application programming interface (API) may be implemented as C code, as a Perl script, as a Java class, or the like, as is described below.

Referring back to Figure 3, Facilitator 302 preferably is a deamon process that runs directly on the customer web or application server. The basic function of the Facilitator is to be queried by instances of the ClientAPI and to handle all of the complexities of fetching the answers. Facilitator, in particular, is responsible for several functions. First, upon starting up, Facilitator retrieves a list of optimal Datad processes to which to connect. This retrieval may be accomplished using a global load balancing infrastructure or any other convenient mechanism provided by the CDN. Facilitator, in a preferred embodiment, retrieves IP addresses of a given number of Datad processes that have (or can be shown to have) good connectivity to the

Facilitator. The Facilitator preferably makes a connection to each of the returned Datad servers (or to some given subset), and it may then use a selection algorithm to elect an optimal Datad server, e.g., by setting preferred priorities based on average response times from each Datad server. Second, the Facilitator is responsible for requesting authentication and authorization, as well as securing communication between the content provider application and the CDNSP network. The authentication and authorization step preferably involves the Facilitator sending a license certificate to the Datad process to which it is connecting. If the license is successfully authenticated, a negotiation process between the Facilitator and the Datad process begins to secure communications over the network, namely, the transfer of a given database response (if the database is not available locally and/or not current) as well as to facilitate the periodic transfers of the database update(s). Third, once the Facilitator has established a good connection to a given Datad process, the Facilitator begins listening on a configured port for requests (preferably over UDP) being made by the ClientAPI. If communications between the ClientAPI and the Facilitator are not secure, they should reside on the same server.

The Datad preferably is a daemon process that runs on the CDNSP network or some other location network location remote from the Facilitator. As described above, a Datad process also responds to queries from a customer application (through the API and Facilitator) if the Facilitator cannot service the query locally. Generalizing, these servers possess databases of CDN-compiled information that may be queried through the service/ delivery system of the invention. In one embodiment, the Datad daemon is a multi-threaded application that receives and maintains TCP connections with multiple Facilitators. The Datad daemon authenticates connection requests from Facilitators by verifying the license certificate received from each Facilitator. The license certificate sent by the Facilitator to the Datad process preferably is a signed certificate using the Digital Signature Algorithm (DSA) or the like. The Datad process responds to different types of data requests from a Facilitator. A first type of request is a query for information about a specific IP address. This type of request is delivered, for example, when the Facilitator is unable to service the request using a local (Facilitator-side) instance of the database. A second type of request is a request for a most current data file (i.e., a new instance of the database, or some portion thereof) that can then be used by Facilitator for localized data lookup. If the Datad server process has a more current data file then the querying Facilitator, the data file is sent over a secure TCP connection to the Facilitator. This is the database update function.

In operation, as the Facilitator connects to and preferably maintains connectivity with one or more Datad processes, it is able to easily fail over to a good connection if a primary Datad

process begins to fail. Also, as the Facilitator preferably has a weighting and prioritization capability, preferably it will always provide the preferred connection to the optimal Datad process. In the unlikely event that no Datad processes are available to the Facilitator, the ClientAPI returns a default answer based on a timeout parameter set in the API.

Integration with a content provider's Web site or other application is straightforward. Preferably, the API is integrated into the provider's Web application to field user requests to the Web site. The API submits a query to the engine for the IP address resolution. Upon receiving the output from the engine, the API parses the response to give the provider access to any of all of the metrics returned from the query. The engine functions to retrieve the queries from Web applications enhanced with the API to conduct the IP address lookup and subsequent return of user access information. Further, the engine maintains the integrity of the database and facilitates the connectivity to the CDNSP or other network for retrieving the most up-to-date database instance. It also controls the backup functionality of direct connectivity to the database servers in the event the local server becomes unavailable or is not the provider's preferred configuration.

The present invention has many advantages. CDNs generate comprehensive databases that map IP addresses (or IP address blocks) to their corresponding geographic region of origin. With ongoing content delivery services, comprehensive data checking activities, and continual map extension activities, the inventive service offers a highly accurate database that is updated frequently. From installation, providers can start managing their content distribution with data about user access including country of origin, region within a country (such as state), the network origin of the user request, such as AOL or AT&T, and the network type such as cable modem, DSL, dial-up or ISDN. The API and engine provide optimized performance and reliability. They are easily integrated in Web application systems as well as custom provider-developed applications that utilize current Web design standards. Preferably, the invention is implemented as a managed service that directly connects to the CDN. This direct connectivity gives the provider access to the most up-to-date information that the CDN has available, as well as access to a backup data source if local database connectivity becomes unavailable.

The present invention has many applications, several of which are identified below. The database is a knowledge base that enables network and content providers to more intelligently serve content and control assets. The invention leverages the CDN's distributed network topology, vast reach (thousands of servers in hundreds of networks worldwide) and intelligent mapping technology to gather data about the end-user's geographic location and network point of origin. As part of the content delivery service, the CDN constantly maps the Internet and

collects useful real-time data. This network mapping data is made available through the invention to network and content providers who wish to improve the planning, design, maintenance and monitoring of their network assets and/or the effectiveness and targeting of their Web site content. From network topography to localized content publishing to digital asset distribution to ad serving control, the invention provides an ideal source of geo-location and last mile bandwidth information based on knowledge of users' access points to the Web. The database preferably covers all assigned, route-able addresses in the commercial Internet Protocol (IP) address space.

The applications for this data include, without limitation:

- Web site content customization
  - Regional specific content delivery - events, promotions, contact info, news...
  - Ad serving based on user location
  - Language translation
  - Regional specific spelling and units of measure
  - Currency conversions

- eCommerce opportunity management
  - Delivering eCommerce sales offers appropriate to different audiences based on location
  - Localized pricing
  - Regional specific contracts and legal documents

- Improved website navigation
  - Removal of splash screens or other query pages asking users to submit their location

- Territory management of content
  - Digital rights management applications
  - Control content presentation based on user location
  - Restrict access to media based on user location

- Broadband content control
  - Deliver rich media to users with the proper access capability

- Web reporting analysis
  - data can be matched with web reporting logs to provide enhanced geo- and bandwidth information on visitors to a site

- Network Planning and Design
-
- Proximal Routing Applications

- Denial of Service Attack Agent Location Analysis

Preferably, the system does not make any attempt to collect any personally identifying information. Rather, the information delivery system provides anonymous information about how a user accesses the Internet, not user-specific information that identifies a person. Information preferably is reported once per visit. The inventive technique does not download anything to the client nor does it attempt to collect anything from a client. Preferably, the invention does not track client usage or history between during or between sessions. Indeed, with the widespread use of DHCP and Firewalls, IP addresses generally do not provide an effective approach to tracking the behavior of an individual user or machine.

Implementation Details:

The remainder of the description details the technical elements of a preferred embodiment of the invention. This section contains a high-level description of the system operation, the API for the client, and instructions as to the behavior.

The invention preferably is provided through three subsystems:

- *Datad* runs on the CDNSP network, preferably at a collocation facility that is near the customers who will use those servers. These machines preferably get full databases that include all the possible information that will be queried, and they can handle requests from multiple customers. A Datad machine verifies the authenticity of clients, and it ensures secure communication.

- *Facilitator* is preferably a daemon that runs directly on the web server, or if not possible, on a machine within the firewall of the system. The Facilitator machine can run in multiple modes, to be explained below, but its basic function is to be queried and to handle all the complexities of fetching the answers. It passes a license to the Datad, and it establishes multiple encrypted channels over which it will try to optimally get answers to queries. It also sets up a UDP listening port on which clients can pull data.

- *ClientAPI* specifies how a content provider server talks to the Facilitator. This API can be provided as source code, explanation of protocol, or dynamic objects that can be used. The API can also be implemented as a program that is used in CGI scripts.

A site administrator preferably installs and runs the Facilitator on each of the site's web servers. If this is not possible, the administrator installs the Facilitator on a dedicated machine that is within the site's firewall. Preferably the Facilitator is not installed on an open machine.

The Database Daemon (Datad)

The database daemon preferably executes on the CDN or other third party network. This process is responsible for several functions:

- *Responding to queries*: Queries are normally generated by a Facilitator. A connection is established (secure, authentic), and depending on the license that a customer is issued, different data maybe sent down based on query. The entire communication preferably is handled over TCP, using a protocol described in the following section.
- *Update data*: The daemon also regularly contacts the CDN's update servers, and it may query to find out if new database information is available. This may include more accurate maps, and possibly more information for any particular IP address.
- *Log / Report load*: The daemon is also responsible for accumulating information about the kind of queries being made, and preferably it makes summaries available through a query function, as well as logging this data to a statistics file.

The daemon preferably achieves the lookups by downloading map files and database files. The map file is a lookup from an IP address to a database entry point (an integer). The database preferably maps the entry point to a record that contains all known data about this IP block, preferably, e.g., Berkeley DB. This server also executes a process that is responsible for making sure the Datad gets restarted if necessary (such as a failure or reboot).

Facilitator

As noted above, the Facilitator is responsible for being the interface the clients use to fetch the data. The Facilitator is configured upon installation and operates in several modes:

- *Enterprise Mode*: In this mode, the Facilitator has a database locally stored. When a query comes in, it looks up in the database and returns the result.
- *CDNPlatform Mode* : In this mode, the Facilitator maintains secure connections open to the CDN through which it transmits queries.
- *Hybrid Mode*: In this mode, the Facilitator sets up a local Facilitator that has the database stored locally but then regularly gets updates to the data from the CDN. This way, whenever the data changes the Facilitator downloads updates. Response times are fast, however, because all the data is local.

CDNPlatform mode behavior

Facilitator preferably resolves a special domain name to receive a list of IP addresses (for the Datad process) that it should contact. In a representative embodiment, this list is set up on the CDN DNS servers to pick the correct set of machines for this client. Upon receiving the list, a TCP connection is opened to each Datad server (or to a given subset), and negotiations for an encryption key start. This includes transmission of given license information. At the same time, a listener thread is created that listens on a specific UDP port for API queries that come in. When a query arrives (using the UDP protocol), it is parsed. Then, intelligence is applied to communicate this query to a queue associated with the best TCP connection. Each queue is dealt with appropriately, pending queries will be sent through a connection (preferably using TCP protocol) and when a response arrives, it is sent back to the client, preferably using the UDP

protocol. Preferably all queries are responded to, and no timeouts are included here. However, when a TCP connection gets slow or has problems, it will not be used. As TCP connections get bad or die, the Facilitator is responsible for refreshing, fixing, or, if appropriate, giving up on them.

ClientAPI

To allow for the most flexibility with systems, the client API is simply specified, preferably as a UDP-based query-response system, with a timeout. Whenever a query needs to be initiated, the querying entity sets up a socket to communicate with the Facilitator, forms a query packet (based on the UDP protocol), and sends it to the server. It then polls the socket for a response and either gets it in time, or times out (at which point a default action is taken).

Protocols

To support the communication, a protocol is defined for the TCP communication and for the UDP communication. Preferably, TCP is used to contact the CDN servers to allow negotiations, authorization and connection setup to be performed once, and then queries are made afterwards.

A TCP connection first needs to be established. Preferably, the license is used and encryption/authentication is enabled. To do so, a client introduces itself to the server, key negotiation is established (for a secure communication between the two), and options are passed through.

As noted above, preferably UDP is used for the regular query-response that is made with every connection. The querying entity sets up a socket from which it will send a query to Facilitator, and on which it waits for the response.

Client side installation

The content provider preferably runs the Facilitator on each of its Web servers. It may also need to place a hole through its firewall for outgoing TCP connections to some port. The content provider may need to obtain and install the license on each of their machines. In addition, the content provider needs to configure the Facilitator and inform the CDNSP of the IP address of the Facilitator so that the CDN may map it to the best Datad servers.

License

The content provider receives from the CDNSP a license, which preferably specifies information about their operating system, machine configuration, name, expiration date, and the content provider's secret key. This information preferably is signed with the CDNSP's public key.

C API:

The API is used to perform a lookup by calling the cdnsp_ip_lookup function, using the following parameters:

```
cdnsp_ip_lookup(const in_addr_t *addr, char *buffer, size_t *len,
        struct timeval *timeout)
```

The arguments to this function are:

addr	An Internet family address (ipv4) that specifies the address being queried. If that is not available, you may also pass in the 32-bit number.
buffer	A pointer that is set to the buffer which will contain the result of this function.
len	An argument/result. It will contain the length of the buffer on calling, and will return the length of the data on result. Please note that if the buffer is too short, you will get an error.
timeout	Is the amount of time for which the function should block waiting on an answer. If this time expires, buffer will contain the default answer instead, and the return code will indicate so.

The result codes for this function can be any possible value, but constants are used to indicate the meaning. A value of zero is a success code, while any other value is an error.

The resulting buffer will contain the IP resolution answer. This answer preferably is encoded as a list of entries separated via a null character, and terminated with a null character. Multiple values for an attribute are separated by a plus (+) character. Each entry is an attribute/value pair separated by an equals sign.

To get a particular attribute's value, a convenience function is provided. This function is not required, but it is useful for retrieving a specific attribute regularly (e.g. country_code) using the following parameters:

```
cdnsp_attr_get(const char *attr, const char *buffer, const size_t buffer_len, char **result)
```

The arguments to this function are:

attr	A null-terminated string indicating the name of the attribute desired.
buffer	A pointer to the result buffer returned from akamai_ip_lookup
buffer_len	A length for the buffer (as was passed in).
result	A pointer to the address in the buffer where the value for the attribute is specified (right after the equals sign). If the attribute does not exist, the result will point to NULL.

The return codes here match that of cdnsp_ip_lookup.

Perl API:

The API is used to perform a lookup in the following manner:

$buffer = cdnsp_ip_lookup(<ip>, <timeout>)

The arguments to this function are:

ip	An Internet family address (ipv4) that specifies the address being queried. If that is not available, you may also pass in the 32-bit number.
timeout	Is the amount of time (in ms) for which the function should block waiting on an answer. If this time expires, buffer will contain the default answer instead, and the return code will indicate so.

The resulting buffer will contain the IP resolution answer. This answer preferably is encoded as a list of entries separated via a null character, and terminated with a null character. Multiple values for an attribute are separated by a plus (+) character. Each entry is an attribute/value pair separated by an equals sign.

To get a particular attribute's value, a convenience function is provided. This function is not required, but it is useful for retrieving a specific attribute regularly (e.g. country_code) using the following parameters:

$country = cdnsp_attr_get(buffer, <attr>)

The arguments to this function are:

attr	A null-terminated string indicating the name of the attribute desired.
buffer	The buffer filled by cdnsp_ip_lookup ($buffer in the above example)

Java API:

To use the Java API, create a new object of class CdnData, using appropriate Java class files. For every lookup to be performed, create a new object of class CdnData, as in the following example:

CdnData JavaDemo = new CdnData("1.2.3.4", 100);

The instantiator accepts the following:

CdnData(String, int)	- As in the example from above
CdnData(String)	- Similar to the first example, except the default timeout value is used
CdnData (InetAddress, int)	- Instead of string, use an InetAddress class
CdnData (InetAddress)	- Similar to the above, but the default timeout is used
CdnData (byte[], int)	- Instead of a string object, pass in a 4 byte

array which represents the IP address in network byte order.

CdnData (byte[]) - Similar to the above, but the default timeout is used.

Given an object (like 'JavaDemo' above), perform the following:

String result = JavaDemo.getAttribute("country_code");

which would assign to "result" the country_code associated with object JavaDemo (which was instantiated with a particular IP address, in this case country_code for IP Address 1.2.3.4). If no such attribute exists, result will be null.

To pull out all attribute/value pairs, use {object}.results, which is a two dimensional array of strings, where:

JavaDemo.results[x][0] is the attribute, and
JavaDemo.results[x][1] is the value associated with the attributes.

For purposes of load balancing or fail over between multiple Facilitators, several additional functions, set_cdn_server(String ip), get_cdn_server(), set_cdn_port(int port), get_cdn_port(), and set_cdn_server_and_port(String ip, int port) are provided. The argument to set_cdn_server is the IP address of a new Facilitator against which to perform queries; the argument to set_cdn_port is the Port of the new Facilitator against which to perform queries. If the functions are able to successfully set the IP address or Port of the new Facilitator, they return *true*; otherwise they retain the IP or Port of the current Facilitator, and return *false*. The set_cdn_server_and_port function sets both the IP and the Port of the new Facilitator, and it retains both the IP and the Port of the current Facilitator if it fails to set either. The get_cdn_server function returns the IP address of the current Facilitator; the get_cdn_port function returns the Port of the current Facilitator. These functions are used for all API versions.

The following provides representative metrics from the database.

Country Code:

CODE	COUNTRY
AD	ANDORRA
AE	UNITED ARAB EMIRATES
AF	AFGHANISTAN
AG	ANTIGUA AND BARBUDA
AI	ANGUILLA ...

Region (State) Code:

Regions Codes when Country Code = US
AL Alabama

AK Alaska
AZ Arizona
AR Arkansas
CA California
CO Colorado ...

Region Codes when Country Code = CA
AB Alberta
BC British_Columbia
MB Manitoba ...

Network Code:

Code	Description
aol	America On-line
att	AT&T
compuserve	Compuserve
earthlink	Earthlink ...

Network Type:

Network type code
dialup
isdn
cable
dsl

DMA References:

DMA : Cities	: States
500 : Portland-Auburn	: ME,NH
501 : New York	: CT,NJ,NY,PA
502 : Binghamton	: NY
503 : Macon	: GA
504 : Philadelphia	: DE,NJ,PA
505 : Detroit	: MI ...

MSA and PMSA References:

MSA : States	: Cities
0040 : TX	: Abilene
0060 : PR	: Aguadilla
0120 : GA	: Albany
0160 : NY	: Albany-Schenectady-Troy
0200 : NM	: Albuquerque
0220 : LA	: Alexandria
0240 : PA	: Allentown-Bethlehem-Easton
0280 : PA	: Altoona ...

PMSA : States	: Cities
1120 : MA-NH	: Boston
1200 : MA	: Brockton
2600 : MA	: Fitchburg-Leominster
4160 : MA-NH	: Lawrence
4560 : MA-NH	: Lowell ...

Time Zone References

Zone Code	GMT Offset	Geographic Area
EST	GMT-5	Eastern standard time
CST	GMT-6	Central standard time
MST	GMT-7	Mountain standard time
PST	GMT-8	Pacific standard time
EST+1	GMT-4	Puerto Rico, Virgin Islands, APO/FPO (Central America)
GMT+1	GMT+1	APO/FPO (Central Europe)

Throughput Codes:

Throughput code	
vhigh	greater than 300 KB/s
high	greater than 128 KB/s, less than 300 KB/s
med	greater than 56 KB/s, less than 128 KB/s
low	less than 56 KB/s

Representative software and hardware configurations are as follows. For the Facilitator server (assuming 512 MB RAM and 500 MB free disk space): Sun Solaris 2.6 or above running on Sun Ultra 5; Linux Red Hat 6.1 or above running on Pentium 233 Mhz PC; FreeBSD 3.4 or above running on Pentium 233 Mhz PC. For the ClientAPI (assuming 64 MB RAM): SunSolaris 2.6 or above running on Sun Ultra 5; Linux Red Hat 6.1 or above running on Pentium 233 Mhz PC; FreeBSD 3.4 or above running on Pentium 233 Mhz PC, server; Microsoft Windows NT 4.0 running on Pentium 233 Mhz PC. The preferred software requirements for the API: Windows – Microsoft Virtual Studio 6 or Borland C++ Builder 5; Java – JDK 1.2 or above; Perl – Perl 5.0 or above. The daemon runs on commodity PC hardware running Linux OS.

Having thus described our invention, what we now claim is set forth below.

CLAIMS

1. An information retrieval system for providing a content provider with geo-location and network information about how users access the Internet, the content provider operating an execution environment in which an application executes, comprising:

an instance of a database supporting the geo-location and network information indexed by IP address block;

an application programming interface (API) integrated into the application for processing an end user request;

code executing in the content provider execution environment and responsive to the API for retrieving given data from the database and returning the given data to the API for use by the application in formulating a custom response to the end user request; and

a database update process executing on a third party network for managing delivery to the content provider execution environment of updated instances of the database.

2. The information retrieval system as described in Claim 1 wherein the geo-location and network information includes information consisting of at least one of the following metrics: a country code, a state or region code of countries, a network origin of a user's access, a network type, a Designated Marketing Area (DMA), a Metropolitan Statistical Area (MSA), a Primary Metropolitan Statistical Area (PMSA), longitude, latitude, FIPS, and time zone.

3. The information retrieval system as described in Claim 1 wherein the code executing in the content provider execution environment includes code for retrieving a list of one or more database update processes.

4. The information retrieval system as described in Claim 3 wherein the code executing in the content provider execution environment includes code for establishing a secure connection to one or more of the database update processes.

5. The information retrieval system as described in Claim 1 wherein the third party network is a content delivery network.

6. The information retrieval system as described in Claim 1 wherein a given database update process maintains a secure connection with a set of one or more content provider execution environments.

7. The information retrieval system as described in Claim 1 wherein the instance of the database is stored in the content provider execution environment.

8. The information retrieval system as described in Claim 1 wherein the API is implemented in a given programming language.

9. In a content provider execution environment in which an application receives end user requests, the improvement comprising:

an instance of a database supporting geo-location and network information indexed by IP address block;

an application programming interface (API) integrated into the application for processing an end user request; and

code (a) for obtaining a list of one or more remote database processes; (b) for establishing and maintaining a secure connection to at least one remote database process; and (b) for responding to API queries to the database.

10. The improvement as described in Claim 9 wherein the geo-location and network information includes information consisting of at least one of the following metrics: a country code, a state or region code of countries, a network origin of a user's access, a network type, a Designated Marketing Area (DMA), a Metropolitan Statistical Area (MSA), a Primary Metropolitan Statistical Area (PMSA), longitude, latitude, FIPS, and time zone.

11. An information retrieval system for providing a content provider with geo-location and network information about how users access the Internet, the content provider operating an execution environment in which an application executes, comprising:

an instance of a database of geo-location and network information supported in the content provider execution environment;

a set of database update processes executing in an execution environment remote from the content provider execution environment;

code executing in the content provider execution environment and that is responsive to a given request for (a) determining whether given geo-location and network information can be retrieved from the database, (b) retrieving the given information from the database if possible; and (c) requesting the given information from a given one of the set of database update processes

if the given information cannot be retrieved from the instance of the database supported in the content provider execution environment.

12. The information retrieval system as described in Claim 11 wherein the code establishes a secure connection to a given database update process and passes given licensing information.

13. The information retrieval system as described in Claim 11 wherein the code establishes a secure connection to each of a plurality of the database update processes in the event a given database update process fails.

14. The information retrieval system as described in Claim 11 further including an application programming interface (API) integrated into the application for generating the given request.

15. An information retrieval system for providing each of a set of content providers with geo-location and network information about how their users access the Internet, each content provider operating an execution environment in which an application executes, comprising:

an instance of a database of geo-location and network information supported in each content provider execution environment;

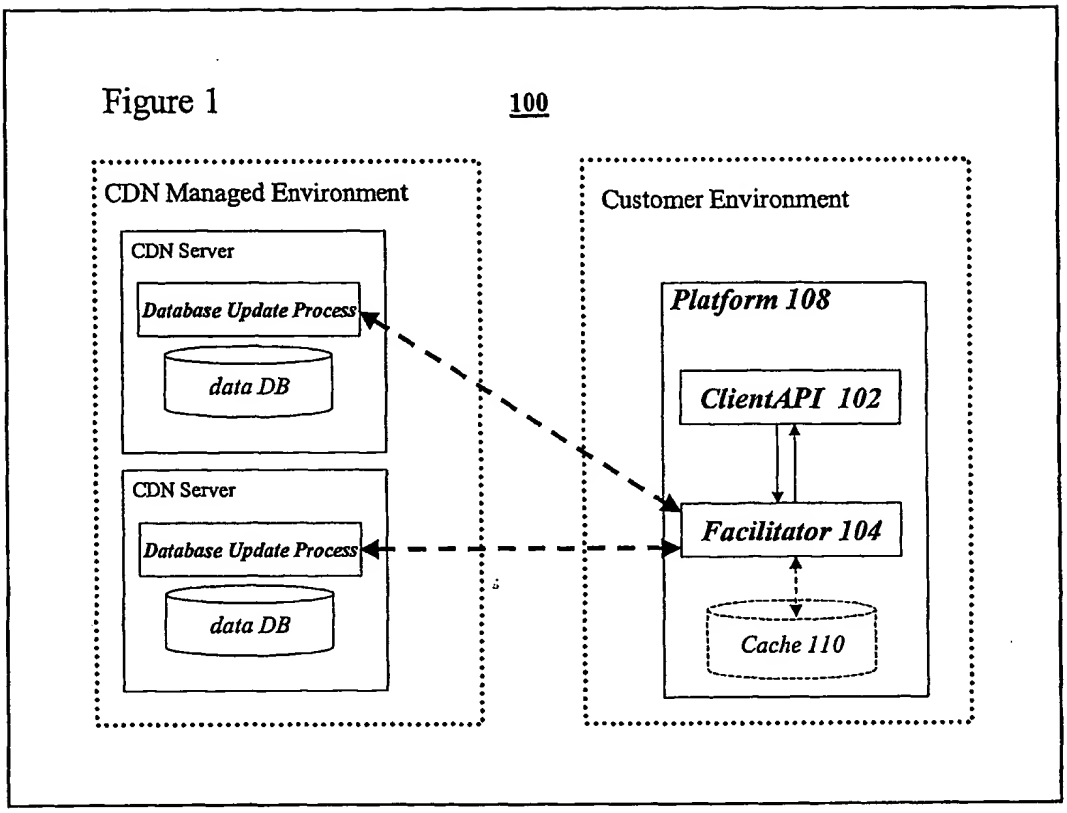
a set of database update processes executing in an execution environment remote from the content provider execution environments; and

an engine executing in each content provider execution environment and that is responsive to a given request for (a) determining whether given geo-location and network information can be retrieved from the database, (b) retrieving the given information from the database if possible; and (c) requesting the given information from a given one of the set of database update processes if the given information cannot be retrieved from the instance of the database supported in the content provider execution environment;

wherein the engine executing in each content provider execution environment has the capability to establish and maintain a secure connection to one or more database update processes and at least one database update process has the capability to maintain a secure connection to one or more engines operating in different content provider execution environments.

16. The information retrieval system as described in Claim 15 further including an application programming interface (API) integrated into the content provider's application for generating the given request.

Figure 1


100
CDN Managed Environment
CDN Server
Database Update Process
data DB
CDN Server
Database Update Process
data DB
Customer Environment
Platform 108
ClientAPI 102
Facilitator 104
Cache 110

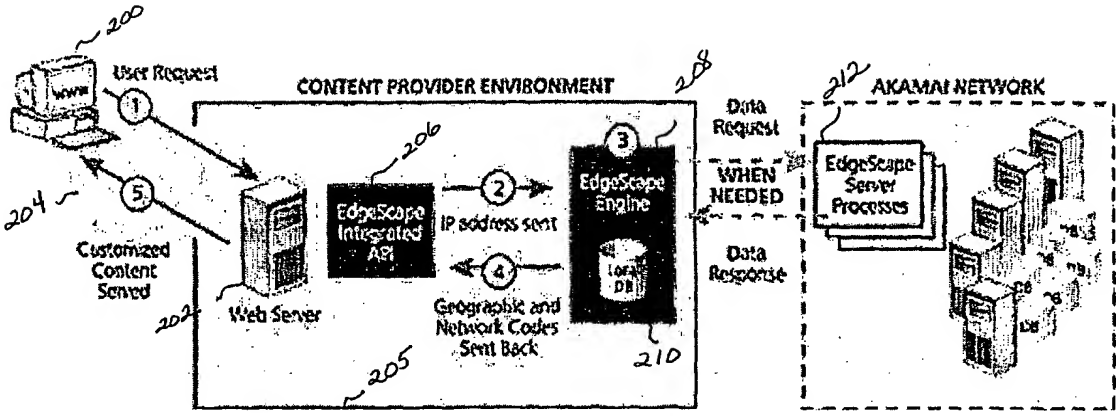
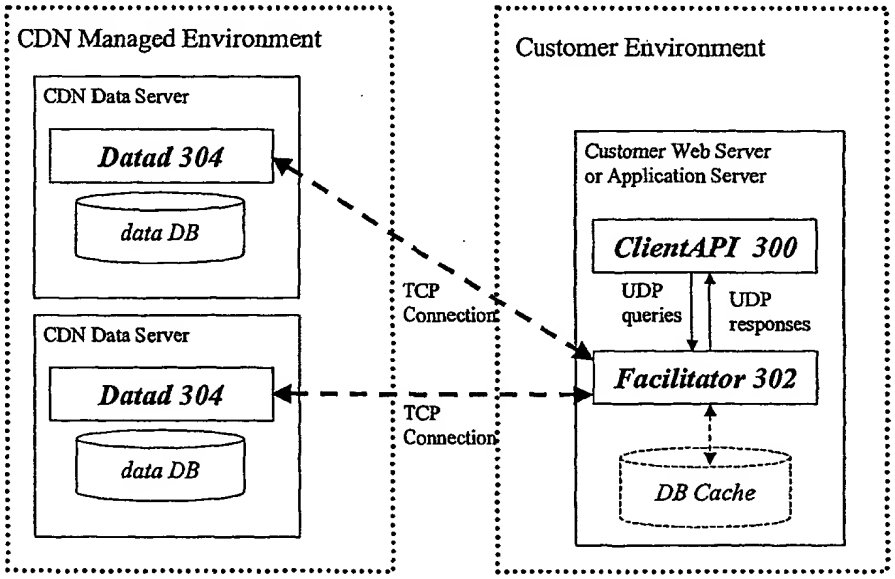


Figure 2

Figure 3



INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/25491

A. CLASSIFICATION OF SUBJECT MATTER
IPC(7) : G06F 17/30
US CL : 707/10, 3
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 707/10, 3, 104.1

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EAST

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y, P	US 6,243,706 B1 (SONDUR et al) 05 June 2001 (05.06.2001), ALL.	1-16
Y, P	US 6,272,343 B1 (PON et al) 07 August 2001 (07.08.2001), ALL.	1-16

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
19 November 2001 (19.11.2001)	14 DEC 2001
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703)305-3230	Authorized officer Uyen T Le Telephone No. 703-305-3900

Form PCT/ISA/210 (second sheet) (July 1998)